2010 24th IEEE International Conference on Advanced Information Networking and Applications

Genetic Algorithms for Efficient Placement of Router Nodes in Wireless Mesh Networks

Fatos Xhafa
Department of Languages and Informatics Systems
Technical University of Catalonia
Campus Nord, C/Jordi Girona 1-3
08034 Barcelona, Spain
EMail: fatos@lsi.upc.edu

Christian Sánchez
Department of Languages and Informatics Systems
Technical University of Catalonia
Campus Nord, C/Jordi Girona 1-3
08034 Barcelona, Spain
EMail: csanchez@lsi.upc.edu

Leonard Barolli
Department of Information and Communication Engineering
Fukuoka Institute of Technology
3-30-1 Wajiro-higashi, Higashi-ku
Fukuoka 811-0295, Japan
EMail: barolli@fit.ac.jp

***Abstract*—In Wireless Mesh Networks (WMNs) the meshing architecture, consisting of a grid of mesh routers, provides connectivity services to different mesh client nodes. The good performance and operability of WMNs largely depends on placement of mesh routers nodes in the geographical area to achieve network connectivity and stability. Thus, finding optimal or near-optimal mesh router nodes placement is crucial to such networks. In this work we propose and evaluate Genetic Algorithms (GAs) for near-optimally solving the problem. In our approach we seek a two-fold optimization, namely, the maximization of the size of the giant component in the network and that of user coverage. The size of the giant component is considered here as a criteria for measuring network connectivity. GAs explore the solution space by means of a population of individuals, which are evaluated, selected, crossed and mutated to reproduce new individuals of better quality. The fitness of individuals is measured with respect to network connectivity and user coverage being the former a primary objective and the later a secondary one. Several genetic operators have been considered in implementing GAs in order to find the configuration that works best for the problem. We have experimentally evaluated the proposed GAs using a benchmark of generated instances varying from small to large size. In order to evaluate the quality of achieved solutions for different possible client distributions, instances have been generated using different distributions of mesh clients (Uniform, Normal, Exponential and Weibull). The experimental results showed the efficiency of the GAs for computing high quality solutions of mesh router nodes placement in WMNs.**

***Index Terms*—Mesh Wireless Networks, Genetic Algorithms, Size of Giant Component, User Coverage, Genetic Operators.**

I. Introduction

Wireless Mesh Networks (WMNs) [1], [9] are currently attracting a lot of attention from wireless research and technology community due to their importance as a means for providing cost-efficient broadband wireless connectivity. Moreover, development in WMNs is being pushed by the ever increasing need in developing and deploying medical, transport and surveillance applications in urban areas, metropolitan, neighboring communities and municipal area networks [3].

WMNs are based on mesh topology, in which every node (representing a server) is connected to one or more nodes, enabling thus the information transmission in more than one path. The path redundancy is a robust feature of this kind of topology. Compared to other topologies, Mesh topology needs not a central node, allowing networks based on such topology to be self-healing. These characteristics of networks with mesh topology, make them very reliable and robust networks to potential server node failures.

In WMNs mesh routers provide network connectivity services to mesh client nodes. Mesh routers are similar to normal routers but incorporate also additional functions to support mesh networking, and are usually equipped with multiple interfaces to work with different wireless technologies. Another feature of this type of routers with respect to usual ones is their ability to provide the same coverage with much less transmitter power through multi-hop communications.

The good performance and operability of WMNs largely depends on placement of mesh routers nodes in the geographical area to achieve network connectivity, stability and user coverage. The objective is to find an optimal and robust topology of the mesh network to support intelligent telecommunication services to clients such as adaptive and flexible wireless Internet access, mobile data, voice and video in addition to supporting other preferred client services.

Unfortunately, node placement problems are shown to be computationally hard to solve to optimality [2], [5], [7], [14], and therefore heuristic and meta-heuristic approaches are the *de facto* approach to solve the problem for practical purposes. Several heuristic approaches are found in the literature for

1550-445X/10 $26.00 © 2010 IEEE
DOI 10.1109/AINA.2010.41

node placement problems in WMNs [4], [8], [10], [11], [13].

In this work we propose and evaluate Genetic Algorithms (GAs) [6] for near-optimally solving the problem. GAs are evolutionary algorithms that try to implement the selection process in nature. GAs start from an initial population of individuals, i.e. feasible solutions of the problem, each having associated a fitness value that indicates how good it is as compared to the rest of individuals. Thus, just as in nature, there are natural processes of selection, reproduction and mutation, GA goes through a similar process of evaluation, selection, crossover, mutation and replacement yielding to the next generation of individuals. The process is repeated through a number of generations during which the best features of parents are passed on to offsprings and thus individuals of better quality are eventually obtained.

In our GA approach we seek a two-fold optimization, namely, the maximization of the size of the giant component in the network and user coverage. The size of the giant component is considered thus as a main criteria for network connectivity. The quality of individuals is thus measured with respect to network connectivity and user coverage, being the former a primary objective. Several genetic operators have been considered in implementing GAs in order to find the configuration that works best for the problem. We have experimentally evaluated the proposed GAs using a benchmark of generated instances varying from small to large size. In order to evaluate the quality of achieved solutions for different possible client distributions, instances have been generated using different distributions of mesh clients (Uniform, Normal, Exponential and Weibull).

The rest of the paper is organized as follows. In Section II we present the definition of the mesh router nodes placement problem in WMNs. The GA's features are briefly introduced in Section III and their application to mesh router nodes placement in Section IV. The experimental evaluation is given in Section V. We end the paper in Section VI with some conclusions.

II. Problem Definition

In a general setting, location models in the literature have been defined as follows. We are given:

- a universe U, from which a set C of client input positions is selected;
- an integer, $N \geq 1$, denoting the number of facilities to be deployed;
- one or more metrics of the type $d : U \times U \rightarrow R_+$, which measure the quality of the location, and
- an optimization model.

Then, the optimization model takes in input the universe where facilities are to be deployed, a set of client positions and returns a set of positions for facilities that optimize the considered metrics.

It should be noted that different models can be established depending on whether the universe is considered: (a) *continuous* (universe is a region, where clients and facilities may be placed anywhere within the *continuum* leading to an uncountably infinite number of possible locations); (b) *discrete* (universe is a discrete set of predefined positions); and, (c) *network* (universe is given by an undirected weighted graph; in the graph, client positions are given by the vertices and facilities may be located anywhere on the graph).

We consider the version of the mesh node placement problem corresponding to the network space model above. Thus, in this version, we are given a 2D area where to distribute a number of mesh router nodes and a number of mesh client nodes of fixed positions (of an arbitrary distribution) and finds a location assignment for the mesh routers that maximizes the network connectivity (size of the giant component) and client coverage. An instance of the problem consists thus of:

- N mesh router nodes, each having its own radio coverage, defining thus a vector of routers.
- An area $W \times H$ where to distribute N mesh routers. Positions of mesh routers are not pre-determined. The area is divided in square cells of an *a priori* fixed length and mesh router nodes are to be deployed in the cells of the grid area.
- M client mesh nodes located in arbitrary cells of the considered grid area, defining a matrix of clients.

An instance of the problem can be formalized by an adjacency matrix of the WMN graph, whose nodes are of two types: router nodes and client nodes and whose edges are links in the mesh network (there is a link between a mesh router and mesh client if the client is within radio coverage of the router). Each mesh node in the graph is a triple $v =< x, y, r >$ representing the 2D location point and r is the radius of the transmission range. There is an arc between two nodes u and v, if v is within the transmission circular area of u. It should be noticed here that the deployment area is partitioned by grid cells, representing graph nodes, where we can locate mesh router nodes. In fact, in a cell, both a mesh and a client node can be placed.

The objective is to place mesh router nodes in cells of considered area to maximize network connectivity and user coverage. Network connectivity and user coverage are among most important metrics in WMNs. The former measures the degree of connectivity of the mesh nodes while the later refers to the number of mesh client nodes connected to the WMN. Both objectives are important and directly affect the network performance; nonetheless, network connectivity is considered as more important than user coverage. It should also be noted that in general optimizing one objective could affect the other objective although there is no direct relation among these objectives nor are they contradicting.

A. Optimization setting

For optimization problems having two or more objective functions, two settings are usually considered: the hierarchical and simultaneous optimization. In the former, the objectives are classified (sorted) according to their priority. Thus, for the two objective case, one of the objectives, say f_1, is considered as primary objective and the other, say f_2, as secondary one. The meaning is that the optimization is carried out in two

steps: in the first we try to optimize f_1, and then, we try to optimize f_2 without worsening the best value of $f1$. In the later approach, both objectives are optimized simultaneously.

In this work we have considered the hierarchical approach in which the size of the giant component is a primary objective and the user coverage is a secondary one. Thus, GAs use this optimization scheme when evaluating the fitness of individuals. In such approach, the network connectivity (through the maximization of size of giant component) is considered as most important since connectivity of the network is crucial for WMNs.

B. Client mesh nodes distributions

It should be noticed from the above problem description that mesh client nodes can be arbitrarily situated in the given area. For evaluation purposes, it is interesting, however, to consider concrete distributions of clients. For instance, it has been shown from studies in real urban areas or university campuses that users (client mesh nodes) tend to cluster to hotspots. Therefore different client mesh nodes distributions should be considered, for instance Weibull distribution, in evaluating WMN metrics.

We have considered Uniform, Normal, Exponential and Weibull distributions for client mesh nodes in the experimental evaluation (see Section V).

III. Genetic Algorithms

GAs have shown their usefulness for the resolution of many computationally combinatorial optimization problems. They are of course a strong candidate for efficiently solving mesh router node placement problem in WMNs. For the purpose of this work we have used the *template* given in Alg. 1.

Algorithm 1 Genetic Algorithm template

Generate the initial population P^0 of size μ;
Evaluate P^0;
while not termination-condition **do**
  Select the parental pool T^t of size λ; $T^t := Select(P^t)$;
  Perform crossover procedure on pairs of individuals in T^t with probability p_c; $P_c^t := Cross(T^t)$;
  Perform mutation procedure on individuals in P_c^t with probability p_m; $P_m^t := Mutate(P_c^t)$;
  Evaluate P_m^t ;
  Create a new population P^{t+1} of size μ from individuals in P^t and/or P_m^t ;
  $P^{t+1} := Replace(P^t; P_m^t)$
  $t := t + 1$;
end while
return Best found individual as solution;

We briefly present next the main features of GAs (see Fig. 1 for its classification in the tree of search methods.)

- *Population of individuals:* Unlike local search techniques that construct a path in the solution space jumping from one solution to another one through local perturbations, GAs use a population of individuals giving thus the search a larger scope and chances to find better solutions. This feature is also known as "exploration" process in difference to "exploitation" process of local search methods.
- *Fitness:* The determination of an appropriate adaptive function or objective function, together with the chromosome encoding are crucial to the performance of GAs. Ideally we would construct objective functions with "certain regularities", i.e. objective functions that verify that for any two individuals which are close in the search space, their respective values in the objective functions are similar.
- *Selection:* The selection of individuals to be crossed is another important aspect in GAs as it impacts on the convergence of the algorithm. Several selection schemes have been proposed in the literature for selection operators trying to cope with premature convergence of GAs.
- *Crossover operators:* Use of crossover operators is one of the most important characteristics. Crossover operator is the means of GAs to transmit best genetic features of parents to offsprings during many generations of the evolution process.
- *Mutation operators:* These operators intend to improve the individuals of a population by small changes. They aim to provide a component of randomness in the neighborhood of the individuals of the population.
- *Escape from local optima:* GAs have the ability to avoid falling prematurely into local optima and can eventually escape from them during the search process.
- *Convergence:* The convergence of the algorithm is the mechanism of GAs to reach to good solutions. A premature convergence of the algorithm would cause that all individuals of the population being similar in their genetic features and thus the search would result ineffective and the algorithm getting stuck into local optima. Maintaining the diversity of the population is therefore very important to this family of evolutionary algorithms.

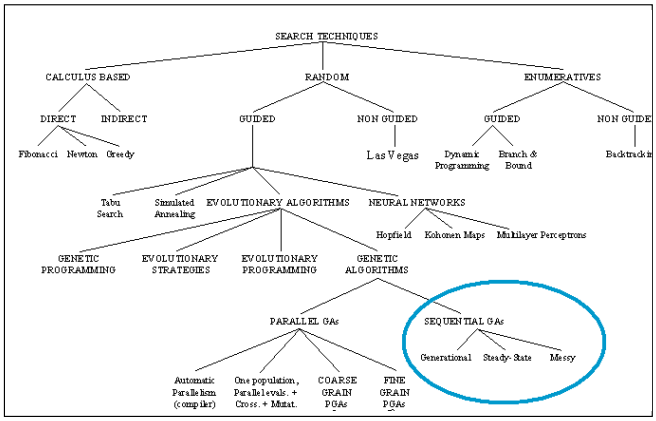

Fig. 1. Classification of GAs in the tree of search methods.

It should be noted however that GAs are computationally expensive algorithms due to heavy use of genetic operators and usually require a large number of iterations to reach

high quality solutions. In particular, as we will see, crossover operator can be computationally expensive for the problem of mesh router nodes placement due to the encoding of solutions in a two-dimensional grid.

IV. GA FOR MESH ROUTER NODE PLACEMENT PROBLEM

We present in this section the particularization of GAs for the problem of mesh router nodes placement in WMNs.

A. Encoding

The encoding of individuals (also known as chromosome encoding) is fundamental to the implementation of GAs in order to efficiently transmit the genetic information from parents to offsprings.

In the case of the mesh router nodes placement problem, a solution (individual of the population) contains the information on the current location of routers in the grid area as well as information on links to other mesh router nodes and mesh client nodes. This information is kept in data structures, namely, `pos_routers` for positions of mesh router nodes, `routers_links` for link information among routers and `client_router_link` for link information among routers and clients (matrices of the same size as the grid area are used.) Based on these data structures, the size of the giant component and the number of users covered are computed for the solution.

It should be also noted that routers are assumed to have different radio coverage, therefore to any router are linked a number of clients and other routers. Obviously, whenever a router is moved to another cell of then grid area, the information on links to both other routers and clients must be computed again.

B. Fitness evaluation

The fitness function is of particular importance in GAs as it guides the search towards most promising areas of the solution space. Furthermore, in our case, we face an optimization problem with multiple criteria, including size of giant component and number of users covered (but could include others such as minimization of number of routers to deploy) and therefore the fitness function in our particular case can be expressed in different ways. In the multi-criteria optimization two most common approaches are the hierarchical and simultaneous approaches.

In the hierarchical optimization approach is one in which we establish a priority (hierarchy) between the different criteria so that if objective A is of higher priority than objective B, then we first optimize objective A and next we optimize objective B but without worsening the value achieved for A (see also Subsec. II-A). This approach is useful when it is needed to prioritize a certain criteria to ensure quality of service (QoS). By contrast, the simultaneous approach considers several objective at the same time; one simple case of the simultaneous approach is the weighted sum of criteria, in which each objective is given a certain weight. However, it is not always possible to express a weighted sum of criteria since criteria may not always be summed up. In fact this is the case of our bi-objective approach, namely, the size of the giant component cannot be summed up with the number of users covered as the two objectives are expressed using different unit measures.

We have thus adopted the bi-objective case, in which the size of the giant component is considered primary and the number of users covered is considered secondary.

C. Selection operators

In the evolutionary computing literature we can find a variety of selection operators, which are in charge of selecting individuals for the pool mate. The operators considered in this work are those based on *implicit fitness re-mapping* technique. It should be noted that selection operators are generic ones and do not depend on the encoding of individuals.

a) Linear Ranking Selection: This operator follows the strategy of selecting the individuals in the population with a probability directly proportional to its fitness value. This operator clearly benefits the selection of best endowed individuals, which have larger chances of being selected.

b) Best Selection: This operator selects the individuals in the population having higher fitness value. The main drawback to this operator is that by always choosing the best fitted individuals of the population, the GA converges prematurely.

c) Tournament Selection: This operator selects the individuals based on the result of a tournament among individuals. Usually winning solutions are the ones to better fitness value but individuals of worse fitness value could be chose as well, contributing thus to avoiding premature convergence. Particular cases of this operator are the Binary Tournament and $N-$Tournament Selection.

D. Crossover operators

The crossover operators are the most important ingredient of GAs. Indeed, by selecting individuals from the parental generation and interchanging their *genes*, new individuals (descendants) are obtained. The aim is to obtain descendants of better quality that will feed the next generation and enable the search to explore new regions of solution space not explored yet.

There exist many types of crossover operators explored in the evolutionary computing literature. It is very important to stress that the crossover operators depend on the chromosome representation. This observation is especially important for the mesh router nodes problem, since in our case, instead of having strings we have a grid of nodes located in a certain position. The crossover operator should thus take into account the specifics of mesh router nodes encoding. We have considered the following crossover operators, called *intersection operators*, which take in input two individuals and produce in output two new individuals (see Alg. 2)

E. Mutation operators

The mutation operator is crucial for preventing the search from getting stuck in local optima by doing small local

Algorithm 2 Crossover operator.

1: **Input**: Two parent individuals P_1 and P_2; values H_g and W_g for height and width of a small grid area;
2: **Output**: Two offsprings O_1 and O_2;
3: Select at random a $H_g \times W_g$ rectangle RP_1 in parent P_1. Let RP_2 be the same rectangle in parent P_2;
4: Select at random a $H_g \times W_g$ rectangle RO_1 in offspring O_1. Let RO_2 be the same rectangle in offspring O_2;
5: Interchange the mesh router nodes: Move the mesh router nodes of RP_1 to RO_2 and those of RP_2 to RO_1;
6: Re-establish mesh nodes network connections in O_1 and O_2 (links between mesh router nodes and links between client mesh nodes and mesh router nodes are computed again);
7: **return** O_1 and O_2;

perturbations to the individuals of the population. Again, the definition of the mutation operators is specific to encoding of the individuals of the concrete problem under study. We defined thus several specific mutation operators as follows:

d) SingleMutate: Select a mesh router node in the grid area and move it to another cell of the grid area (see Fig. 2 (left)). After the move is done, network connections are computed again.

e) RectangleMutate: This operator selects two "small" rectangles at random in the grid area, and swaps the mesh routers nodes in them. Certainly, in this case the modification of the individual is larger than in the case of SingleMutate (see Fig. 2 (right)).

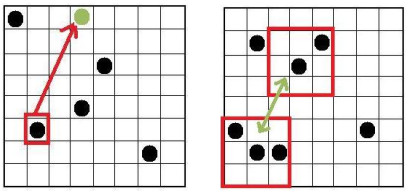

Fig. 2. Mutation operators: SingleMutate (left) and RectangleMutate (right).

f) SmallMutate: This operator chooses randomly a router and moves it a small (*a priori* fixed) numbers of cells in one of the four directions: up, down, left or right in the grid.

g) SmallRectangleMutate: This operator is similar as SmallMutate but now we select first at random a rectangle and then all routers inside the rectangle are moved with a small (*a priori* fixed) numbers of cells in one of the four directions: up, down, left or right in the grid.

Again, after the mutation is done, network connections (the links between routers and links between routers and users) are re-computed.

V. Experimental study

A. Parameter setup

Parameter setup is a main issues in effectively using heuristic approaches since parameter values have a direct impact on the performance of the algorithm. As usually, parameters are classified into two groups: parameters related to the heuristic method itself, the GA in our case, and parameters related to the problem under study, the mesh router node placement in our case.

h) GA parameters: In this group we have the following parameters: population size, intermediate population size, number of evolution steps, crossover probability, mutate probability and parameters for replacement strategies such as replace only if better or generational replacement.

i) Mesh router node placement parameters: In this group we have the number of routers to deploy, number of client nodes to cover, grid area sizes, etc.

The fine tuning of parameters is known for its complexity due to the large numbers of parameters as well as due to possible synergies and side effects among different parameters values. On the other hand, the values of these parameters should be set up independently and in a way that they are effective for any instances of the problem, although tuning will be conducted using a selected sample of instances. To this end, randomly generated instances of three different grid area sizes (32x32, 64x64 and 128x128, respectively) are used. To avoid biased results, 15 independent runs of GA were performed. Then, the resulting setting of parameter is used for obtaining computational results for a benchmark of instances.

To exemplify the tuning process, we present next the results for mutate operators for instances of 32x32, 64x64 and 128x128 grid area sizes.

j) Instances of 32x32 grid area size: In this case the setting of parameters obtained is: cross probability=0.8, population size=26, intermediate population size=12 and mutate probability=0.2. In the instances, the client positions were generated following a normal distribution $N(\mu = 16, \sigma = 32/10)$, and 16 routers were to be placed in the 32x32 grid area to cover 48 clients.

The averaged results of 15 independent runs showed that for small size instances the mutation operator `SingleMutate` performed best (see Fig. 3).

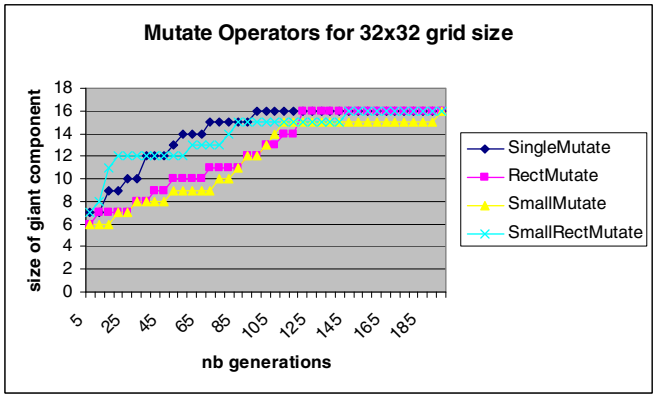

Fig. 3. Performance of mutate operators in GA algorithm for 32x32 grid area where 16 routers were to be placed and give coverage to 48 clients.

k) Instances of 64x64 grid area size: In this case the setting of parameters obtained is: cross probability=0.75, population size=36, intermediate population size=17 and mutate probability=0.25. In the instances, the client positions were generated following a normal distribution $N(\mu = 32, \sigma = 64/10)$, and 32 routers were to be placed in the 64x64 grid area to cover 96 clients.

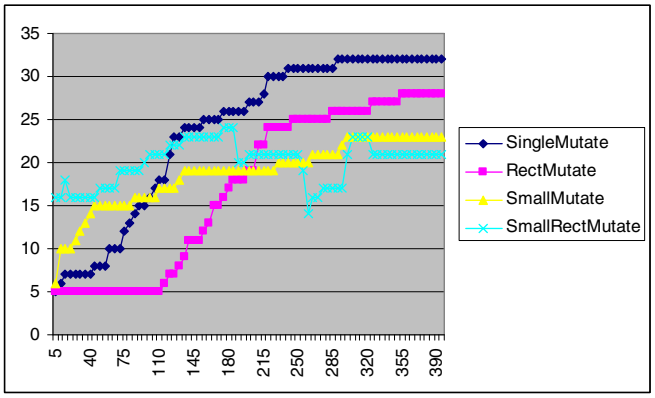

Fig. 4. Performance of mutate operators in GA algorithm for 64x64 grid area where 32 routers were to be placed and give coverage to 96 clients.

As can be seen from Fig. 4, `SingleMutate` showed again the best performance. The RectangleMutate however is also a good candidate.

l) Instances of 128x128 grid area size: In this case the setting of parameters obtained is: cross probability=0.8, population size=49, intermediate population size=24 and mutate probability=0.20. In the instances, the client positions were generated following a normal distribution $N(\mu = 64, \sigma = 128/10)$, and 64 routers were to be placed in the 128x128 grid area to cover 192 clients.

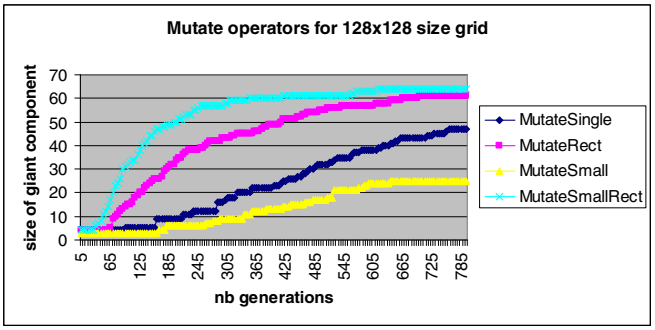

Fig. 5. Performance of mutate operators in GA algorithm for 128x128 grid area where 64 routers were to be placed and give coverage to 192 clients.

As can be seen from Fig. 5, the mutate operators that perform best are those that move several mesh router nodes (in a neighborhood area) at once, that is, MutateRectangle and MutateSmallRectangle.

For large size instances, SingleMutate doesn't perform well as it moves just one mesh router node out of many candidate ones.

B. Tuning of other parameters

The setting of the rest of the parameters is done as follows.

m) start_choice: This parameter indicates the methods used to generate the initial population. Two individuals have been computed using StartNear and StartHotSpot (see [12] for implementation of several *ad hoc* methods); the rest of individuals of the population are generated at random.

n) population_size and intermediate_population_size: As replacement operator was chosen the Elitist Generational, therefore, the setting of the population_size and that of intermediate_population_size satisfies intermediate_population_size = population_size - 2, where population_size is taken of logarithmic order $\Theta(\log_2(N)^k)$, where N is the total number of mesh router nodes.

o) nb_evolution_steps: This parameter indicates the number of generations performed by the algorithm. Its value is fixed according to the size of the input, namely, $5 \cdot size\_grid\_x \cdot \delta$, where δ is a constant for adjusting the resulting value of number of evolutions steps.

p) cross_probability and mutate_probability: These parameters have been set to $p_c = 0.8$ and $p_m = 0.2$, respectively.

C. Benchmark of instances

We have generated a benchmark consisting of 48 instances, having different sizes of grid area and using four probability distributions for the positions of mesh client nodes in the grid area. These instances aim to represent realistic-size instances[1]

Instances are arranged in three groups, each having 16 instances and are labelled $I_{x \times x\_D\_k}$, where:

- x stands for the height and width of the grid area, that is, the number of cells of arbitrary edge length; it takes values 32, 64 and 128.
- D stands for the distribution of the client mesh routers in the grid area; four distributions are considered: Uniform (U), Normal (N), Exponential (E) and Weibull (W).
- k is the index of the instance.

Thus, we have 16 instances for each grid size (32, 64 and 128, resp.) and within each group we have 4 instances for each distribution (Uniform, Normal, Exponential and Weibull, resp). For instance, in this notation, $I_{64 \times 64\_N\_3}$ denotes the third instance of a 64×64 grid area, with mesh clients nodes positions generated using Normal distribution.

Finally, notice that instances of 32×32 grid area consist of 16 mesh routers nodes and 48 client mesh nodes; instances of 64×64 grid area consist of 32 mesh routers nodes and 96 client mesh nodes; and, instances of 128×128 grid area consist of 64 mesh routers nodes and 192 client mesh nodes.

D. Results of GA for the benchmark

Now that we have adjusted the parameters for the three possible sizes of instances, we can run the algorithm on benchmark instances and evaluate the quality of obtained solutions with respect to the four client mesh nodes distributions.

[1]In the literature, instances having up to 60 mesh devices are usually considered realistic-size instances.

TABLE I
SIZE OF GIANT COMPONENT AND USER COVERAGE VALUES FOR 32×32 GRID SIZE INSTANCES, 16 ROUTERS NODES AND 48 CLIENTS.

Instance	size of giant component				Users covered			
	best	avg	dev	ini	best	avg	dev	ini
I32x32_U_1	16	16	0	5	17	16	0.1	14
I32x32_U_2	16	16	0	7	17	16	0.1	9
I32x32_U_3	16	16	0	5	15	14	0.1	10
I32x32_U_4	16	16	0	5	15	14	0.1	14
I32x32_N_1	16	16	0	11	43	40	0.3	25
I32x32_N_2	16	16	0	7	41	40	0.1	21
I32x32_N_3	16	16	0	8	42	41	0.1	20
I32x32_N_4	16	16	0	6	39	38	0.1	24
I32x32_E_1	16	16	0	6	43	43	0	16
I32x32_E_2	16	16	0	6	22	18	0.4	8
I32x32_E_3	16	16	0	7	29	25	0.4	13
I32x32_E_4	16	16	0	6	37	32	0.5	10
I32x32_W_1	16	16	0	6	35	25	1	5
I32x32_W_2	16	16	0	6	30	26	0.4	14
I32x32_W_3	16	16	0	6	30	16	1.4	16
I32x32_W_4	16	16	0	6	31	25	0.6	15

q) Computational results for instances of size 32x32 grid area: We give in Table I computational results for instances of benchmark of 32x32 grid area. In the table, best indicates the best value out of 15 runs, avg –the average value, dev– the deviation, and ini the initial value of the size of the giant component.

As can be seen from Table I, with 200 generations the GA algorithm achieved to establish a network of all routers connected. However, the number of the users covered depends on the distribution of clients in the grid area, achieving the best values for normal distribution of clients. In fact, the algorithm achieved good results in fewer generations (about 120 generations), as shown graphically in Fig. 6. On the other hand the evolution of the number of the users covered can be seen in Fig. 7.

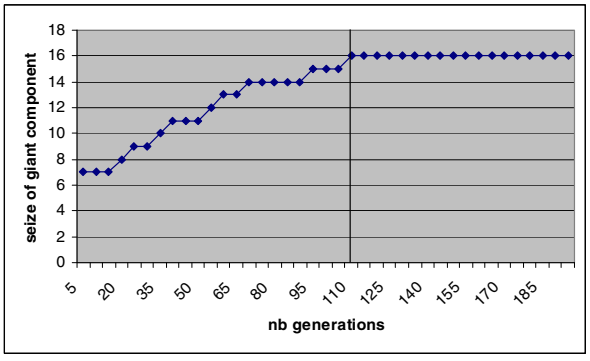

Fig. 6. Evolution of size of giant component in GA algorithm for 32x32 grid area.

r) Computational results for instances of size 64x64: We give in Table II computational results for instances of benchmark of 64x64 grid area.

The evolution of the size of the giant component obtained by GA for 64x64 grid area size is shown in Fig. 8.

s) Computational results for instances of size 128x128 grid area: We give in Table III computational results for

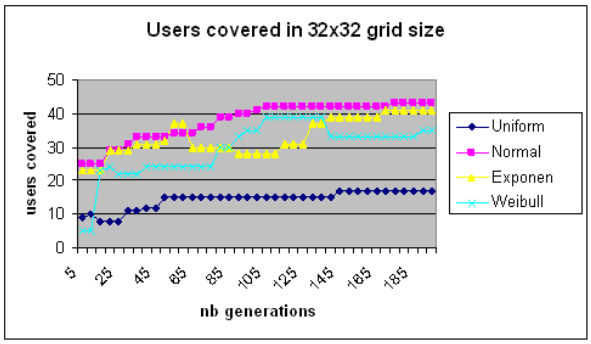

Fig. 7. Evolution of number of users covered in GA algorithm for 32x32 grid area.

TABLE II
SIZE OF GIANT COMPONENT AND USER COVERAGE VALUES FOR 64×64 GRID SIZE INSTANCES, 32 ROUTERS NODES AND 96 CLIENTS.

Instance	size of giant component				Users covered			
	best	avg	dev	ini	best	avg	dev	ini
I64x64_U_1	32	32	0	6	11	8	0.3	13
I64x64_U_2	32	32	0	6	12	10	0.2	10
I64x64_U_3	32	32	0	7	15	15	0	13
I64x64_U_4	32	32	0	8	17	17	0	7
I64x64_N_1	32	32	0	5	64	44	2	24
I64x64_N_2	32	32	0	6	66	55	1.1	18
I64x64_N_3	32	31	0.1	8	59	49	1	24
I64x64_N_4	32	32	0	5	60	50	1	19
I64x64_E_1	32	31	0.1	4	5	3	0.2	24
I64x64_E_2	32	31	0.1	6	10	2	0.2	7
I64x64_E_3	32	31	0.1	6	10	10	0	16
I64x64_E_4	32	32	0	8	5	5	0	8
I64x64_W_1	32	32	0	4	39	32	0.7	22
I64x64_W_2	32	32	0	5	50	42	0.8	39
I64x64_W_3	32	32	0	8	10	10	0	2
I64x64_W_4	32	32	0	8	10	10	0	3

instances of benchmark of 128x128 grid area (see Fig. 9 for the graphical representation).

As can be seen from Table III, the GA algorithm performed very well for all but normal distribution of clients in the grid area.

E. Analysis of the results

From the computational results we can see that the normal and uniform distributions of clients correspond to the best

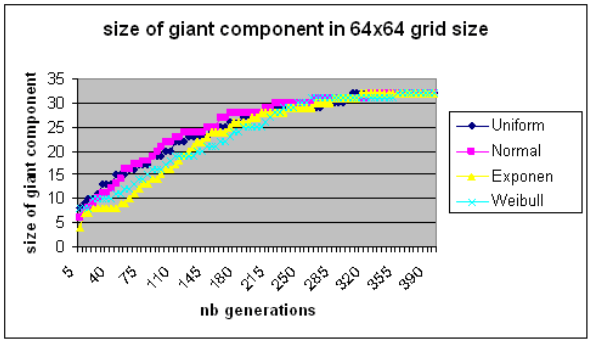

Fig. 8. Evolution of size of giant component in GA algorithm for 64x64 grid area.

TABLE III
SIZE OF GIANT COMPONENT AND USER COVERAGE FOR 128×128 GRID SIZE INSTANCES, 64 ROUTERS NODES AND 192 CLIENTS.

Instance	size of giant component				Users covered		
	best	avg	dev	ini	best	avg	ini
I128x128_U_1	61	55	0.6	5	17	9	12
I128x128_U_2	59	55	0.4	5	18	16	13
I128x128_U_3	63	56	0.7	4	16	10	14
I128x128_U_4	59	55	0.4	4	16	13	11
I128x128_N_1	64	57	0.7	4	70	39	17
I128x128_N_2	62	58	0.4	4	63	42	10
I128x128_N_3	62	57	0.5	4	76	46	22
I128x128_N_4	63	58	0.5	4	70	46	20
I128x128_E_1	59	55	0.4	3	62	39	26
I128x128_E_2	60	54	0.6	4	35	20	19
I128x128_E_3	61	54	0.7	6	40	12	15
I128x128_E_4	60	54	0.6	6	44	24	17
I128x128_W_1	62	54	0.8	4	36	10	12
I128x128_W_2	58	53	0.5	3	45	33	26
I128x128_W_3	61	55	0.6	5	38	34	24
I128x128_W_4	58	54	0.4	4	52	44	18

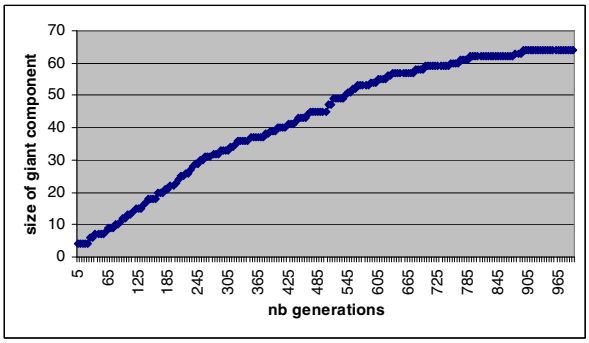

Fig. 9. Evolution of size of giant component in GA algorithm for 128x128 grid area.

results obtained by the GA algorithm. It should be noticed however that the uniform distribution causes premature convergence. On the other hand, from the results we can see that there is no deviation in the size of the giant component, actually the GA achieved almost always connectivity of all mesh router nodes; however, the deviation in the case of number of users covered is considerable. Finally, it's worth observing that the results corresponding to the Weibull and Exponential distributions are rather similar.

VI. CONCLUSIONS

In this work we have presented Genetic Algorithms (GAs) for the problem of mesh router nodes placement in Wireless Mesh Networks (WMNs). In this problem, we are given a number of client mesh nodes *a priori* distributed in a grid area –arranged in small cells– and a given number of mesh router nodes are to be deployed in the cells of the grid area. We have considered the bi-objective optimization in which we want to maximize the network connectivity of the WMN (through the maximization of the size of the giant component) and the maximization of the number of the user coverage (client mesh nodes). In the model, the former objective is considered as primary while the later is considered secondary objective, that is, the algorithm tries to optimize first the size of giant component and then tries to maximize the number of clients covered without worsening the size of the giant component.

The analysis of experimental results showed that GA are very efficient at computing placement of mesh router nodes and almost always achieve to establish connectivity of all mesh router nodes. However, the user coverage is more sensible to the distribution of mesh client nodes in the deployment grid area. The proposed approach has practical usefulness for designing and deploying of real WMNs. In our future work we would like to evaluate the GA placement algorithm under a dynamic environment.

ACKNOWLEDGMENT

Fatos Xhafa's research work partially done at Birkbeck, University of London, on Leave from Technical University of Catalonia (Barcelona, Spain). His research is supported by General Secretariat of Universities of the Ministry of Education, Spain.

REFERENCES

[1] I. F. Akyildiz, X. Wang, and W. Wang. Wireless mesh networks: a survey. Computer Networks **47**(4) (2005) 445-487.

[2] E. Amaldi, A. Capone, M. Cesana, I. Filippini, F. Malucelli. Optimization models and methods for planning wireless mesh networks. Computer Networks **52** (2008) 21592171.

[3] Ch. Chen and Ch. Chekuri. Urban Wireless Mesh Network Planning: The Case of Directional Antennas. Tech Report No. UIUCDCS-R-2007-2874, Department of Computer Science, University of Illinois at Urbana-Champaign, (2007).

[4] A. Antony Franklin and C. Siva Ram Murthy. Node Placement Algorithm for Deployment of Two-Tier Wireless Mesh Networks. In Proceedings of IEEE GLOBECOM 2007, IEEE Global Communications Conference (2007), 4823-4827.

[5] M.R. Garey and D.S. Johnson. Computers and Intractability –A Guide to the Theory of NP-Completeness. Freeman, San Francisco, (1979).

[6] J. Holland. *Adaptation in Natural and Artifitial Systems*. University of Michigan Press, Ann Arbor, (1975).

[7] A. Lim, B. Rodrigues, F. Wang and Zh. Xua. $k-$Center problems with minimum coverage. *Theoretical Computer Science* **332** (2005) 1-17.

[8] S. N. Muthaiah and C. Rosenberg. Single Gateway Placement in Wireless Mesh Networks. In Proceedings of 8th International IEEE Symposium on Computer Networks, Turkey, (2008).

[9] N. Nandiraju, D. Nandiraju, L. Santhanama, B. He, J. Wang, and D. Agrawal. Wireless mesh networks: Current challenges and future direction of web-in-the-sky. IEEE Wireless Communications (2007) 79-89.

[10] M. Tang. Gateways Placement in Backbone Wireless Mesh Networks. International Journal of Communications, Network and System Sciences, **1** (2009) 1-89.

[11] T. Vanhatupa, M. Hännikäinen and T.D. Hämäläinen. Genetic Algorithm to Optimize Node Placement and Configuration for WLAN Planning. In Proceedings of 4th International Symposium on Wireless Communication Systems (2007), 612-616.

[12] F. Xhafa, Ch. Sanchez, L. Barolli. Ad Hoc and Neighborhood Search Methods for Placement of Mesh Routers in Wireless Mesh Networks. In Proceedings of ICDCS Workshops of the IEEE 29th International Conference on Distributed Computing Systems (ICDCS'09) (2009) 400-405.

[13] P. Zhou, B. S. Manoj, and R. Rao. A gateway placement algorithm in wireless mesh networks. In Proceedings of the 3rd international Conference on Wireless internet (Austin, Texas, October 22 - 24, 2007). ICST (Institute for Computer Sciences Social-Informatics and Telecommunications Engineering), ICST, Brussels, Belgium, 1-9.

[14] J. Wang, B. Xie, K. Cai and D.P. Agrawal. Efficient Mesh Router Placement in Wireless Mesh Networks, Proceedings of IEEE MASS'07, (2007) 1-9.